RAIRO Operations Research
RAIRO Oper. Res. **41** (2007) 253–273
DOI: 10.1051/ro:2007023

ON THE MINIMUM COST MULTIPLE-SOURCE UNSPLITTABLE FLOW PROBLEM

Meriema Belaidouni[1] and Walid Ben-Ameur[1]

Abstract. The minimum cost multiple-source unsplittable flow problem is studied in this paper. A simple necessary condition to get a solution is proposed. It deals with capacities and demands and can be seen as a generalization of the well-known semi-metric condition for continuous multicommdity flows. A cutting plane algorithm is derived using a superadditive approach. The inequalities considered here are valid for single knapsack constraints. They are based on nondecreasing superadditive functions and can be used to strengthen the relaxation of any integer program with knapsack constraints. Some numerical experiments confirm the efficiency of the inequalities introduced in the paper.

Keywords. Network flows, integer programming, superadditive functions.

Mathematics Subject Classification. 90C10, 90B18.

Introduction

The minimum cost multiple-source unsplittable flow problem is defined as follows: let $R = (X, E, c)$ be an arc-capacitated network where X is the set of nodes, E the set of arcs, and each arc ij has a capacity c_{ij}. Let K denote the set of commodities. Each commodity k has an origin (source) $o(k)$, a destination (sink) $s(k)$ and a requested flow value d^k. Let us denote by $P(k)$ the set of all possible simple paths for commodity k. Assigning a commodity k to an arc ij costs $w_{ij}^k d^k$, where w_{ij}^k is the unit flow cost for arc ij and commodity k. The problem consists

Received December 1, 2006. Accepted December 21, 2006.

[1] GET/INT - CNRS UMR 5157, Institut National des Télécommunications 9, rue Charles Fourier, 91011, Evry, France; walid.benameur@int.evry.fr

in routing each commodity k through a single-path $p \in P(k)$ such as the total flow through any edge is at most its capacity and the cost function is minimized.

The problem is sometime called the minimum cost single-path routing problem.

Single-path routing is highly appreciated in communication networks. When single-path routing is considered in a packet network, the arriving order of packets is the same as their departure order. This means that we do not need to implement any mechanism to support in-order packet delivery. Moreover, as we need only one path per commodity, the routing tables will have a reasonable size. Finally, the processing time needed to forward traffic will also be lower when single-path routing is considered.

From a pure optimization point of view, there is no doubt that the most efficient way to avoid congestion, for instance, by keeping the maximum load as low as possible, would be to split the traffic freely over all the network. However, it is not difficult to see that an optimal solution of a linear multi-commodity flow problem with known capacities (at least if obtained by a simplex based method, which is most often the case) will use a mean number of paths per demand ranging between 1 and $1+\frac{|E|}{|K|}$ (see, *e.g.*, [9]). For most real communication networks, the number of edges is about $2n$ (two times the number of vertices) and the number of demands is $n(n-1)/2$. It follows that $\frac{|E|}{|K|}$ is usually very small (from 0.2 for a 20 node network up to 0.05 for a 80 node network) and almost all the demands are hence routed on a single-path. Of course, it is possible to build examples for which the difference between multi-path routing and single-path routing is very large (see *e.g.* [9, 28]), but this situation seldom occurs in practical instances.

Unsplittable flow problems have been extensively studied during the last decade. They are generally NP-hard problems. Many combinatorial problems can be easily reformulated as a single-path routing problem (bin packing, the partition problem, some scheduling problems, the knapsack problem, etc.) [17, 24, 35].

An important result related to single-path routing is given in [17]. It is shown in this paper that any single-source multiple-path routing can be transformed into a single-path routing with an increase in terms of link loads (flows) bounded by the value of the maximum demand. Moreover, when the network is a ring, an optimal multiple-source multiple-path routing can be transformed into a single-path routing with an increase in terms of link loads bounded by 3/2 times the maximum demand [35]. A generalization of the algorithm of [35] is provided in [10].

In fact, many other problems related to unsplittable flow problems are defined in [24]. If the network capacities are not sufficient to carry all commodities then one may try to route only a subset of demands and choose them such that the sum of the routed demands is maximum. One may also divide the set of demands in a minimum number of subsets (rounds) such that each set of demands can be routed through the network. This is called the minimum number of rounds problem. A third problem consists in minimizing the network congestion which is defined by the maximum through E of the ratios $\frac{f_{ij}}{c_{ij}}$ where f_{ij} is the traffic flowing on the arc ij. Many approximation algorithms have been proposed for these

three problems (see, *e.g.*, [4,24–26,36]). Some bicriteria approximation algorithms are also proposed in [25, 36] and other papers. For example, when single-source problems are considered, [36] provides a polynomial time algorithm to find a single-path routing such that the congestion is less than $3 + 2\sqrt{2}$ times the optimal congestion and the cost is less than the optimal cost.

A generalization of unsplittable flow problems is studied in [6] where t paths can be used to satisfy a commodity.

Some simple heuristics are proposed in [4, 13, 37] to solve the minimum cost multiple-source flow problem. Several Meta-heuristics are also used to handle the problem [3, 27].

Cutting plane and column generation algorithms have also been used to solve the problem in [2, 7, 18, 32, 33]. More details about these algorithms are given in Section 1.

Our paper deals with exact solution methods. Two cutting plane algorithms based on superadditive cuts are proposed. Notice that the inequalities introduced in this paper are valid for single knapsack constraints. Thus, they can be used to strengthen the relaxation of any integer program with knapsack constraints.

As mentioned above, two cutting plane algorithms are proposed: SCPA and QSCPA. Compared to SCPA, QSCPA is basically a restriction to the special case of step functions. We will prove that the inequalities generated by QSCPA are equivalent to those of SCPA in terms of violation by the current solution.

To evaluate their performances, both algorithms are compared to the branch-and-price-and-cut algorithm of [7]. The computational trials are ran on several instances proposed by [34] and considered by [7]. These instances are arising from a teleconferencing application.

The superadditive approach and the Superadditive Cutting Plane Algorithm (SCPA) are introduced in Section 2. We will also recall some of the known results related to superadditivity.

QSCPA is described in Section 3. A generalization of the well known Japanese theorem of [23, 31] is proposed in Section 4 where a simple necessary condition to the Minimum Cost Single-Path Routing Problem is provided. While this condition was not really used in the derivation of the cutting plane algorithm, it is one of the main contributions of the paper.

Computational experiments are reported in Section 5. Finally, a conclusion and some further research directions are provided in Section 6.

1. Some column generation and cutting plane algorithms

First, let us give the problem formulation. The minimum cost multiple-source unsplittable flow problem can be described using either a node-arc formulation or a path formulation. In what follows, we recall both formulations (see, *e.g.*, [1])

Node-arc formulation

$$(\mathcal{NAF}) \begin{cases} \min \sum_{k\in K} \sum_{ij\in E} w_{ij}^k d^k x_{ij}^k & \\ \sum_{k\in K} d^k x_{ij}^k \leq c_{ij} & \forall ij \in E \\ \sum_{ij\in E} x_{ij}^k - \sum_{ji\in E} x_{ji}^k = b_i^k & \forall i \in X, \forall k \in K \\ x_{ij}^k \in \{0,1\} & \forall ij \in E, \forall k \in K \end{cases} \tag{1}$$

where $x_{ij}^k = 1$ if the requested flow value d^k is assigned to arc ij and equals 0 otherwise. The first set of inequalities are the capacity constraints. They express the requirement that the total flow on each arc cannot exceed the arc capacity. The second set of inequalities are flow conservation constraints for each node where $b_i^k = 1$ if i is the source of commodity k, $b_i^k = -1$ if i is the destination node of commodity k, and $b_i^k = 0$ otherwise. The objective function is a linear function representing the total cost to be minimized.

Path formulation

$$(\mathcal{PF}) \begin{cases} \min \sum_{k\in K} \sum_{p\in P(k)} w_p^k d^k y_p^k & \\ \sum_{k\in K} \sum_{p\in P(k)} d^k y_p^k \delta_{ij}^p \leq c_{ij} & \forall ij \in E \\ \sum_{p\in P(k)} y_p^k = 1 & \forall k \in K \\ y_p^k \in \{0,1\} & \forall k \in K, \forall p \in P(k) \end{cases} \tag{2}$$

where $y_p^k = 1$ if the requested flow value d^k is assigned to path p and equals 0 otherwise. Notice that we used here δ_{ij}^p which is equal to 1 if and only if the path p is using the arc ij. The constraints are quite straightforward and do not need any comment. We only have to remember that this kind of formulation is solved using column generation where columns correspond to paths. They are generated by shortest path computations where the edge weights are the values of the dual variables (see, *e.g.*, [1, 7] for more details).

The node-arc formulation and the path formulation are equivalent: the costs are positive so all the used paths do not contain cycles and any optimal solution of $(\mathcal{NAF})$ leads to an optimal solution of $(\mathcal{PF})$ and *vice-versa* (see, *e.g.*, [1]). Note However that if we consider a fractional solution of $(\mathcal{NAF})$ (corresponding to a multiple-path routing), it is generally NP-hard to find exactly a minimum number of paths to decompose the flow (see [12] for more details).

Barnhart *et al.* [7] solved the minimum cost multiple-source unsplittable flow problem using a branch-and-price-and-cut algorithm. They used a path formulation. The cutting planes considered in [7,18] are lift and cover cuts. Given an arc ij, if a set of demands C is such that $\sum_{k\in C} d^k > c_{ij}$ then

$$\sum_{k\in C} x_{ij}^k \leq |C| - 1$$

is clearly a valid inequality. If the path formulation is used, the cover inequality can be trivially expressed using the flow path variables. The cover inequalities that are not dominated by other cover inequalities are called minimum cover inequalities. In this case, in addition to $\sum_{k\in C} d^k > c_{ij}$, we should have $\sum_{k\in C'\subsetneq C} d^k \leq c_{ij}$ for any $C' \subsetneq C$. Cover inequalities can be generated by solving a knapsack problem [7]. They may be lifted in a classical way to get stronger inequalities and tighter relaxations. However, as pointed out by the authors of [7] and analyzed in details by Gu *et al.* [20,21], generating these cuts and finding the best among them may be very time consuming. Many details about the implementation of the branch-and-price-and-cut are given in the paper [7]: how should we branch such that the pricing problem remains easy to solve; how cuts are added to improve the relaxation without destroying the nice feature of pricing.

Surveys about Branch-and-price-and-cut are given in [8,29,38]. Other cutting plane algorithms have been proposed in [18,32]. They are based on the same kind of cuts as those used in [7]. The branching strategies and the heuristics used to generate cuts are the main differences between the papers [7,18,32].

A slightly different routing problem has been studied in [16]. The model of [16] includes an intermediate layer and assumes that the traffic demand values are equal to either 1 or a constant value B. A flow formulation is considered and a polyhedral study is provided and used in a cutting plane algorithm.

A network design problem where single-path routing is considered and link capacities are integer valued, is studied in [5]. The sum of an installation cost and a routing cost are minimized in this paper. Some families of valid inequalities incorporating the routing variables and the capacity variables (capacities are variables in [5]) are provided and some separation algorithms are described.

Recently, another approach has been proposed in [2,33]. It is based on a lagrangian relaxation of the classical flow conservation constraints. The dual problem becomes a knapsack problem. In other terms, a particular Dantzig-Wolfe decomposition called knapsack decomposition is used to solve the problem. To be more precise, the convex hull of the $0-1$ solutions of the knapsack problem $\sum_{k\in K} d^k x_{ij}^k \leq c_{ij}; x_{ij}^k \in \{0,1\}, \forall k \in K$ can be described using its extreme points

denoted by $z_{ij_1}, z_{ij_2}, \ldots$. Said another way, $\sum_{k\in K} d^k x^k_{ij} \leq c_{ij}$ can be replaced by

$$\begin{aligned} x_{ij} &= \sum_t \lambda_t z_{ij_t} \\ \sum_t \lambda_t &= 1 \quad , \lambda_t \geq 0 \end{aligned}$$

where x_{ij} is the vector whose components are the x^k_{ij}. The problem can be solved if we have an efficient procedure to generate the extreme points z_{ij_t}. One can easily show that this can be done by solving a $0-1$-knapsack problem. While the column generation problem is NP-hard, the computational results of [2, 33] seem to be good. Notice that the knapsack decomposition has been previously used in other papers (see, for instance, [14, 22]).

2. A superadditive approach

Superadditive cuts

A function $F : D \subset R^m \to R$ is called *superadditive* over D if

$$F(v_1) + F(v_2) \leq F(v_1 + v_2) \quad \forall v_1, v_2, v_1 + v_2 \in D. \tag{3}$$

Note that $v_1 = 0$ yields $F(0) + F(v_2) \leq F(v_2)$ or $F(0) \leq 0$. We assume in this paper that $F(0) = 0$ and $0 \in D$.

The functions considered in this paper to generate cuts are not only superadditive but also nondecreasing. In other terms,

$$v_1 \geq v_2 \Rightarrow F(v_1) \geq F(v_2) \quad \forall v_1, v_2 \in D. \tag{4}$$

Proposition 2.1 (see, *e.g.*, [30]). *If $F : R^m \to R$ is superadditive and nondecreasing then*

$$\sum_{j=1}^{j=n} F(a_j)x_j \leq F(b) \tag{5}$$

is a valid inequality for $S = Z^n \cap \{x \in R^n_+ : Ax \leq b\}$, for any (A, b), a_j is the jth column of A.

The inequality given by (5) is called *superadditive valid inequality* or *superadditive cut* for short.

Theorem 2.2 (see, *e.g.*, [11, 30]). *Every valid inequality for a nonempty $S = Z^n \cap \{x \in R^n_+ : Ax \leq b\}$ is equivalent to or dominated by a superadditive cut.*

Generation of superadditive cuts (Gomory theorem)

Let $S = \{x \in Z_+^n, Ax \leq b\}$ such that all coefficients of (A, b) are nonnegative integers and let $\mathcal{D}(b) = \{v \in Z_+^m : v \leq b\}$. We assume that $b_i \geq \max\{a_{ij}$ for all $j\}$ implying that the columns of the matrix A are in $\mathcal{D}(b)$. All maximal valid inequalities of S (other than $x \geq 0$) can be obtained by a superadditive nondecreasing function $F : \mathcal{D}(b) \to [0, 1]$ satisfying the linear inequalities of the polytope (P).

$$(P)\begin{cases} F(v_1) + F(v_2) - F(v_1 + v_2) & \leq & 0 & \forall v_1, v_2 \in \mathcal{D}(b), v_1 + v_2 \leq b \\ F(v) & \geq & 0 & \forall v \in \mathcal{D}(b) \\ F(b) & = & 1. & \end{cases} \tag{6}$$

Any function F satisfying the constraints (6) will produce a valid inequality for the set $conv(S)$. Notice that the functions defined by (P) will be nondecreasing on $\mathcal{D}(b)$.

Many other results related to superadditivity and duality are reported in [30]. Most of these results focus on general integer programs. Wolsey provided in [39] some results related to $0-1$ integer programs where he showed that valid inequalities can be characterized by two underlying functions, one of which is superadditive. More recent results are given in [19].

The aim of this paper is to use the Gomory theorem to build an efficient cutting plane algorithm to solve the minimum cost multiple-source unsplittable flow problem. The inequalities that will be generated are generally not facets of the corresponding polyhedron.

The superadditive cutting plane algorithm (SCPA)

The heart of SCPA is to derive superadditive cuts from knapsack constraints. Given a single capacity constraint $\sum_r a_r x_r \leq b$ and an optimal fractional solution x^* for the relaxed single-path routing problem, we intend to find a nondecreasing superadditive function $\bar{F}$ belonging to (P) and maximizing $\sum_r x_r^* \bar{F}(a_r) - \bar{F}(b)$. This program is noted (LP). Using an optimal function $\bar{F}$ and the single capacity constraint $\sum_r a_r x_r \leq b$, we get the superadditive cut $\sum_r \bar{F}(a_r) x_r \leq \bar{F}(b)$.

The algorithm SCPA aims to generate cutting planes using each single knapsack constraint $\sum_j a_{ij} x_{ij} \leq b_i$ of the node arc formulation $\mathcal{NAF}$.

SCPA is sketched below.

Algorithm 1 SCPA

Step 0: Let $\mathcal{NAF}_r$ be the linear relaxation of the node arc formulation $\mathcal{NAF}$.

Step 1: Solve $\mathcal{NAF}_r$. Let x^* be the current optimal solution of $\mathcal{NAF}_r$.

Step 2: If x^* is fractional go to **Step 3**. Otherwise stop.

Step 3: For each capacity constraint of $\mathcal{NAF}$ (let i be its line number)

Step 3.1: Consider the polytope (P_i) obtained from (P) when b reduces to b_i.

Step 3.2: Find an optimal solution $\bar{F}$ and the optimal value $\bar{\delta}$ to the linear program (LP) given by: Maximize $\sum_j x_j^* F(a_{ij}) - F(b_i)$ on the polyhedron (P_i)

Step 3.3: If $\bar{\delta} > 0$, then add the violated superadditive cut

$$\sum_j \bar{F}(a_{ij}) x_j \leq \bar{F}(b_i)$$

to $\mathcal{NAF}_r$ and $\mathcal{NAF}$.

Step 4: If no superadditive cuts were generated, then stop and apply Branch & Bound to $\mathcal{NAF}$. Otherwise go to **Step 1**.

To speed up SCPA, Step 1 can be carried out using column generation applied to the path formulation $\mathcal{PF}$. The solution y^* of the relaxed path program $\mathcal{PF}_r$ can be translated to x^* (needed to define (LP)) using the following equations:

$$x_{ij}^k = \textstyle\sum_{p \in P(k)} y_p^k \delta_{ij}^p \quad \forall ij \in E. \tag{7}$$

Equations (7) can also be used to translate superadditive cuts from node-arc formulation to path formulation in Step 3.3. Column generation is based on reduced cost consideration and is done in a classical way by shortest path calculations. However, if the solution is not integral, we switch to the node-arc formulation before starting the Branch&Bound. We can of course keep the path formulation during the Branch&Bound but the pricing problem should be handled as done in [7].

Notice that the problem (LP) related to a knapsack constraint $\sum_j a_{ij} x_j \leq b_i$ contains about b_i variables and $O(b_i^2)$ constraints. This clearly means that the separation procedure is generally not polynomial. In other words, SCPA can be time consuming when the right hand sides (the numbers b_i) are large. This led us to propose some improvements.

3. Some improvements

The aim of this section is to introduce the Quick Superadditive Cutting Plane Algorithm (QSCPA).

QSCPA is derived from SCPA. It is based on the following observation: one of the most expensive steps of the algorithm SCPA is the definition of the polyhedron (P_i) (Step 3.1 of the algorithm SCPA).

One way to avoid the description of the polyhedron (P_i) is to focus on some families of superadditive funtions. Some of them are described in the next section.

3.1. Superadditive families

First, let us recall the concept of step functions.

A function $F_l : [0, b] \to [0, 1]$ defined by

$$F_l(x) = \begin{cases} u_i & \text{If} \quad \alpha_i \leq x < \alpha_{i+1} \quad \text{and} \quad 0 \leq i \leq l-1 \\ u_l & \text{If} \quad x = \alpha_l \end{cases} \tag{8}$$

is a step function where $(\alpha_i)_{0 \leq i \leq l-1}$ is a nondecreasing sequence with $\alpha_0 = 0$ and $\alpha_l = b$. A step function F_l can also be noted $(\alpha_i, u_i)_{0 \leq i \leq l}$.

The first family ϕ_l of superadditive functions that we propose presents the following advantages: (*i*) it may be described using linear constraints; (*ii*) the complexity to construct the corresponding polyhedron is $O(l^2)$ (l may be chosen very small even though b is large); (*iii*) any nondecreasing subadditive sequence (α_i) can be used to generate this kind of functions.

The family ϕ_l is the set of all step functions $F_l : [0, b] \to [0, 1]$ where

$$\alpha_0 = 0, \alpha_l = b, \quad u_0 = 0, u_l = 1 \tag{9}$$
$$\alpha_i + \alpha_j \geq \alpha_{i+j}, \quad 0 \leq i \leq l, 0 \leq j \leq l, 0 \leq i + j \leq l \tag{10}$$
$$u_i + u_j \leq u_{i+j}, \quad 0 \leq i \leq l, 0 \leq j \leq l, 0 \leq i + j \leq l \tag{11}$$
$$\alpha_i - \alpha_{i-1} \geq 0, \quad 1 \leq i \leq l \tag{12}$$
$$u_i \geq 0, \quad 0 \leq i \leq l. \tag{13}$$

Notice that if $\alpha_i = \alpha_{i-1}$, then we impose that $u_i = u_{i-1}$ to make sense. In this case, we will necessarily have $u_1 = 0$.

The next propositions states all the functions defined in this way are superadditive.

Proposition 3.1. *All the functions in ϕ_l are superadditive and nondecreasing.*

Proof. $F_l \in \phi_l$ is clearly nondecreasing. Let x and y be two reals in $[0, b]$ such that $x + y \leq b$ and let i and j be the greatest integers such that $x \geq \alpha_i$ and $y \geq \alpha_j$. Thus $\alpha_i + \alpha_j \leq x + y \leq b$. Moreover, $b = \alpha_l \leq \alpha_i + \alpha_{l-i}$. This leads to $\alpha_j \leq \alpha_{l-i}$. □

Two cases will be studied: $j > l - i$ and $j \leq l - i$.

Suppose that $j > l - i$. Since α is a non decreasing sequence, we must have $\alpha_j = \alpha_{l-i}$ and $u_j = u_{l-i}$. This leads to $x = \alpha_i$ and $y = \alpha_j = \alpha_{l-i} = b - \alpha_i$. Using (11) we get $F_l(x + y) = F_l(b) \geq u_i + u_{l-i} = F_l(x) + F_l(y)$.

Let us now focus on the second case: $j \leq l - i$. We can use again (11) to deduce that $u_{i+j} \geq u_i + u_j = F_l(x) + F_l(y)$. $x + y \geq \alpha_i + \alpha_j \geq \alpha_{i+j}$ implies that $F_l(x + y) \geq u_{i+j}$ (we use again the fact that F_l is nondecreasing). Combining the previous two inequalities leads to $F_l(x + y) \geq F_l(x) + F_l(y)$.

Remark

- Links with Gomory Theorem: observe that all functions F that satisfy the inequalities of (P) belong also to the family $\phi_{l=b}$. We only have to take:

$$l = b, \alpha_i = i, u_i = F(i) \quad \forall i \in \mathcal{D}(b). \tag{14}$$

A second family ψ_l of superadditive functions is proposed below. It presents the following advantages: (*i*) it is more general than the previous family ϕ_l;(*ii*) it is partially described using linear constraints; (*iii*) it can be easily generated by choosing any increasing sequence α.

ψ_l is the set of step functions $F_l : [0, b] \to [0, 1]$ where

$$\alpha_0 = 0, \alpha_l = b, \quad u_0 = 0, u_l = 1 \tag{15}$$
$$\alpha_{i+1} - \alpha_i \geq 0, \quad 0 \leq i \leq l-1 \tag{16}$$
$$u_{i+1} - u_i \geq 0, \quad 0 \leq i \leq l-1 \tag{17}$$
$$u_i \geq 0, \qquad 0 \leq i \leq l \tag{18}$$

$$\alpha_i + \alpha_j \leq b \Rightarrow \exists k / \begin{cases} \alpha_i + \alpha_j \geq \alpha_k \\ u_i + u_j \leq u_k \end{cases} \quad 0 \leq i \leq l, 0 \leq j \leq l. \tag{19}$$

Notice that if $\alpha_i = \alpha_{i-1}$, then we impose that $u_i = u_{i-1}$ to make sense.

Proposition 3.2. *The functions in ψ_l are superadditive and nondecreasing.*

Proof. Let x and y be two reals such that $x + y \leq b$ and let i and j be the greatest integers such that $x \geq \alpha_i$ and $y \geq \alpha_j$. Thus $x + y \geq \alpha_i + \alpha_j$. Using (19) we get $x + y \geq \alpha_k$. It follows from the definition of a step function, equations (17) and (18), that F_l is nondecreasing. Thus $F_l(x + y) \geq u_k$. Using (19) we obtain that $F_l(x + y) \geq u_i + u_j$. Since $F_l(x) = u_i$ and $F_l(y) = u_j$, we get $F_l(x + y) \geq F_l(x) + F_l(y)$. □

Remark

- The family of superadditive functions ψ_l is more general than the family of superadditive functions ϕ_l. Indeed, let F be a superadditive function of the family ϕ_l. For each couple i, j we define $i + j = k_{i,j}$. It follows $\alpha_{i+j} = \alpha_{k_{i,j}}$ and $u_i + u_j \leq u_{k_{i,j}}$. Thus $F \in \psi_l$.
- Observe that $\psi_l \not\subset \phi_l$. Indeed, Let F_6 be a step function defined as follows: $b = 60$, ($\alpha_0 = 0$,$\alpha_1 = 20$,$\alpha_2 = 25$,$\alpha_3 = 39$,$\alpha_4 = 43$,$\alpha_5 = 47$,$\alpha_6 = 60$), ($u_0 = 0$, $u_1 = 0.33$,$u_2 = 0.5$,$u_3 = 0.67$,$u_4 = 0.83$,$u_5 = 1$,$u_6 = 1$). We have $F_6 \in \psi_l$. But $F_6 \notin \phi_l$ since $u_1 + u_2 > u_3$.

3.2. Quick superadditive cutting plane algorithm (QSCPA)

QSCPA is based on the superadditive family ψ_l. A quadratic problem $(\mathcal{QP}_l)$ is dealt with by fixing some of the variables and solving a linear program.

Separation problem

Given a single knapsack constraint $\sum_r a_r x_r \leq b$ and an optimal solution x^* of the relaxed single-path routing problem, QSCPA aims to find a function $\bar{F}$ from the superadditive family ψ_l such that the objective function $\sum_r x_r^* F(a_r) - F(b)$ is maximized. This program is denoted by $(\mathcal{QP}_l)$.

$$(\mathcal{QP}_l)\begin{cases} \max \sum_j x_j^* F_l(a_j) - F_l(b) & & \\ \alpha_0 = 0, \alpha_l = b, u_0 = 0, u_l = 1 & & \\ \alpha_i + \alpha_j \leq b \Rightarrow \exists k / \begin{cases} \alpha_i + \alpha_j \geq \alpha_k \\ u_i + u_j \leq u_k \end{cases} & 0 \leq i \leq l & 0 \leq j \leq l \\ \alpha_{i+1} - \alpha_i \geq 0 & 0 \leq i \leq l-1 & \\ u_{i+1} - u_i \geq 0, & 0 \leq i \leq l-1 & \\ u_i \geq 0 & 0 \leq i \leq l & \\ (u_i - F_l(a_j))(\alpha_{i+1} - a_j) \geq 0 & 0 \leq i \leq l-1 & 0 \leq j \leq l. \end{cases}$$

Observe that $(\mathcal{QP}_l)$ contains the constraints defining the family (ψ_l) in addition to constraints

$$(u_i - F_l(a_j))(\alpha_{i+1} - a_j) \geq 0, \quad 0 \leq i \leq l-1, 0 \leq j \leq l. \tag{20}$$

Constraints (20) clearly lead to $F_l(a_j) \leq u_i$ when $\alpha_i \leq a_j < \alpha_{i_+1}$. Since $x_j^* \geq 0$, we can assume that $F_l(a_j) = u_i$.

To solve $(\mathcal{QP}_l)$, we linearize its non linear equations by fixing the value of the finite series (sequence) $\{\alpha_i\}_{0 \leq i \leq l}$.

The series $\bar{\alpha}$ that we propose is defined by Algorithm A:

Algorithm A **Generation of the series $\bar{\alpha}$**

Step 0: Let $\sum_{r=1}^n a_r x_r \leq b$ be a knapsack inequality and x^* an optimal fractional solution to the relaxed node arc formulation $\mathcal{NAF}_r$.

Step 1: Let $\mathcal{B}(x^*)$ be the set of columns j associated to non zero values of x^*. $\mathcal{B}(x^*) = \{j \in N : 1 \leq j \leq n, x_j^* \neq 0\}$.

Step 2: Let $\mathcal{D}(x^*, b)$ be the set of all **different** integer values strictly smaller than b which are linear combinations of $\{a_j\}$, $j \in \mathcal{B}(x^*)$. $\mathcal{D}(x^*, b) = \{\sum_{j \in \mathcal{B}(x^*)} n_j a_j : \sum_{j \in \mathcal{B}(x^*)} n_j a_j < b, n_j \in N\}$.

Step 3: Let $l^* = |\mathcal{D}(x^*, b)| + 1$. Define $\bar{\alpha} = \{\bar{\alpha}_i\}_{0 \leq i \leq l^*}$ as follows

$$\bar{\alpha}_i = \begin{cases} i^{th} \text{ smallest value of } \mathcal{D}(x^*, b) & \text{if } 0 \leq i < l^* \\ \alpha_{l^*} = b & \text{if } i = l^* \end{cases} \tag{21}$$

The sequence $\bar{\alpha}$ is chosen in this way to reduce the complexity of the separation problem: instead of considering the whole set $\mathcal{D}(b)$ as done in SCPA, we take only the subset $\mathcal{D}(x^*, b)$. We will prove in Proposition 3.3 that this is not really

a restriction: the cut that will be generated using the sequence $\bar{\alpha}$ has the same violation as the best cut derived from (P).

After fixing the value of the variables α to $\bar{\alpha}$ defined by Algorithm A, the program $(\mathcal{QP}_l)$ reduces to the linear program $(\mathcal{LP}_{x^*}(\bar{\alpha}))$. Notice that $\bar{\alpha}$ associates with each couple (i,j) a unique value k_{ij} such that $\bar{\alpha}_i+\bar{\alpha}_j = \bar{\alpha}_{k_{ij}}$ when $\bar{\alpha}_i+\bar{\alpha}_j \leq b$. The variables of $(\mathcal{QP}_l)$ are $\{u_i\}_{0\leq i\leq l^*}$. Each variable $F_{l^*}(a_j)$ is replaced by its corresponding variable in $\{u_i\}_{0\leq i\leq l^*}$ using the definition of the step function $F_{l^*} = (\bar{\alpha}_i, u_i)_{0\leq i\leq l^*}$.

$$(\mathcal{LP}_{x^*}(\bar{\alpha}))\begin{cases} \max \sum_r x_r^* F_{l^*}(a_r) - F_{l^*}(b) & \\ u_0 = 0, u_{l^*} = 1 & \\ u_i + u_j \leq u_{k_{ij}} & 0\leq i\leq l^*, 0\leq j\leq l^*, \quad \bar{\alpha}_i+\bar{\alpha}_j \leq b \\ u_{i+1} - u_i \geq 0 & 0\leq i\leq l^*-1 \\ u_i \geq 0 & 0\leq i\leq l^*. \end{cases}$$

Let $\bar{u} = \{\bar{u}_i\}_{0\leq i\leq l^*}$ be the optimal solution of $(\mathcal{LP}_{x^*}(\bar{\alpha}))$. Using $\bar{\alpha}$ and $\bar{u}$ we define the superadditive function $\bar{F}_{l^*} = (\bar{\alpha}_i, \bar{u}_i)_{0\leq i\leq l^*}$.

We already noticed that Algorithm A defines a sequence $\bar{\alpha}$ that allows to reduce the complexity of the separation problem. The next proposition states that using this sequence to get a superadditive cut does not introduce any degradation in terms of violation. In other words, given a single knapsack constraint $\sum_r a_r x_r \leq b$ and an optimal fractional solution to the relaxed node arc formulation $\mathcal{NAF}_r$, let $\bar{F}_{l^*}$ be the superadditive function defined using $\bar{\alpha}$ and $\bar{u}$, where $\bar{\alpha}$ is defined by Algorithm A and $\bar{u}$ is an optimal solution of $\mathcal{LP}_{x^*}(\bar{\alpha})$. Then the violation is given by $\bar{\delta} = \sum_r x_r^* \bar{F}_{l^*}(a_r) - \bar{F}_{l^*}(b)$. We also define $\delta = \max_{F\in(P)} \sum_r x_r^* F(a_r) - F(b)$ representing the maximum violation that we can get if we take any superadditive function derived from (P). We will show that $\bar{\delta} = \delta$.

Proposition 3.3.

$$\bar{\delta} = \delta$$

Proof. Let F be a function belonging to (P) such that $\sum_r x_r^* F(a_r) - F(b)$ is maximum over (P). Thus, we have $\delta = \sum_r x_r^* F(a_r) - F(b)$. □

Let $\widehat{F} : \mathcal{D}(x^*, b)\cup\{b\} \to [0,1]$ be the restriction of F to $\mathcal{D}(x^*, b)\cup\{b\}$ (*ie.* $\widehat{F}(v) = F(v), \forall v \in \mathcal{D}(x^*, b)\cup\{b\}$). Since F is superadditive, $\widehat{F}$ is superadditive.

Let $\widehat{\delta} = \sum_r x_r^* \widehat{F}(a_r) - \widehat{F}(b)$. We have $\delta = \widehat{\delta}$.

We associate to $\widehat{F}$ a step function $\widehat{F}_l$, where $l = |\mathcal{D}(x^*, b)| + 1$.

$$\widehat{F}_l(x) = \begin{cases} \widehat{F}(\bar{\alpha}_i) & \text{If} \quad \bar{\alpha}_i \leq x < \bar{\alpha}_{i+1} \quad \text{and} \quad 0\leq i\leq l-1 \\ \widehat{F}(\bar{\alpha}_l) & \text{If} \quad x = \bar{\alpha}_l. \end{cases}$$

The step function $\widehat{F}_l \in \psi_l$. Thus $\widehat{F}_l$ is superadditive.

Let $\widehat{\delta}_l = \sum_r x_r^* \widehat{F}_l(a_r) - \widehat{F}_l(b)$. We have $\widehat{\delta}_l = \widehat{\delta}$. By $\bar{\alpha}$ definition, one can deduce that $\bar{\delta} \geq \widehat{\delta}_l$. Using the previous equalities we obtain $\bar{\delta} \geq \delta$. Moreover, since the restriction on $\mathcal{D}(b)$ of any function of ψ_l is in (P), we have $\bar{\delta} \leq \delta$.

We can now describe the algorithm QSCPA.

QSCPA DESCRIPTION

The description of QSCPA is given by Algorithm 2. As pointed out before, the only difference between SCPA and QSCPA is Step 3. Therefore, only Step 3 is described in what follows.

Algorithm 2 QSCPA Description of Step 3

Step 3: Let x^* be the optimal fractional solution the relaxed node arc formulation $\mathcal{NAF}_r$. For each capacity constraint of $\mathcal{NAF}$ (let i be its line number)

Step 3.1: Generate the series $\bar{\alpha}$ using Algorithm A.

Step 3.2: Find an optimal solution $\bar{u}$ and the optimal value $\bar{\delta}$ to the linear program $(\mathcal{LP}_{x^*}(\bar{\alpha}))$

Step 3.3: If $\bar{\delta} > 0$, then define $\bar{F}_{l^*}$ using $\bar{\alpha}$ and $\bar{u}$ and add the violated superadditive cut

$$\sum_j \bar{F}_{l^*}(a_{ij}) x_j \leq \bar{F}_{l^*}(b_i)$$

to $\mathcal{NAF}_r$ and $\mathcal{NAF}$.

Observe that for each column r such that $x_r^* = 0$ and $a_r \notin \mathcal{D}(x^*, b)$, the value $\bar{F}_{l^*}(a_r)$ is not given by the linear program $(\mathcal{LP}_{x^*}(\bar{\alpha}))$, but is deduced from the definition of the step function $\bar{F}_l$.

Remarks

The algorithm QSCPA proposed in this section is not the only method that can be used to efficiently generate superadditive cuts. We believe that other superadditive families can be defined and used to accelerate the cutting plane algorithm.

Even if we focus on the first superadditive family, one can easily see that the violation $\sum_r x_r^* F_l(a_r) - F_l(b)$ can be maximized using a simple enumeration algorithm to decide to which interval each a_r belongs followed by a linear program with $O(l)$ variables and $O(l^2)$ constraints. This can be useful if l is small.

We can also choose any increasing subadditive series α and solve the corresponding linear program to maximize $\sum_r x_r^* F_l(a_r) - F_l(b)$.

Similarly, any increasing sequence α can be chosen and a simple linear program can be solved to find a violated superadditive function belonging to the second family defined in this section. In other words, the set $\mathcal{D}(x^*, b)$ used to build the

sequence $\bar{\alpha}$ is only an example. One can choose any other increasing sequence. We only have to guarantee that the size of the linear separation problem $\mathcal{LP}_{x^*}(\alpha)$ remains reasonable.

4. A semi-metric condition for the multiple-source unsplittable flow problem

Let $\lambda = (\lambda_1, ..., \lambda_{|E|}) \in R_+^{|E|}$ and $\pi_k(\lambda)$ be the length of the shortest path joining the source to the sink of commodity k in (X, E), when a length $\lambda_{ij} \geq 0$ is assigned to each arc $ij \in E$. A well known result [23,31] states that when the traffic can be split, the multiple-path routing problem has a solution if and only if the following semi-metric conditions are satisfied:

$$\sum_{ij \in E} \lambda_{ij} c_{ij} \geq \sum_{k \in K} \pi_k(\lambda) d^k. \tag{22}$$

Recall that c_{ij} is the value of the capacity of ij and d^k is the demand of commodity k. This result obtained by duality is sometimes called the Japanese Theorem.

We give here a simple necessary condition for single-path routing.

Theorem 4.1. *The minimum cost multiple-source unsplittable flow problem has a solution only if the following superadditive semi-metric cuts are satisfied:*

$$\sum_{ij \in E} \lambda_{ij} F(c_{ij}) \geq \sum_{k \in K} \pi_k(\lambda) F(d^k) \tag{23}$$

where $\lambda = (\lambda_1, ..., \lambda_{|E|}) \in R_+^{|E|}$ and $\pi_k(\lambda)$ are defined as above and F is any nondecreasing superadditive function.

Inequality (23) is directly derived from the validity of the inequalities

$$\sum_{k \in K} F(d^k) x_{ij}^k \leq F(c_{ij}).$$

In fact, these inequalities imply that we can route the demands $F(d^k)$ on the capacities $F(c_{ij})$ for any superadditive nondecreasing function F. Then we can apply the Japanese theorem to say that $\sum_{ij \in E} \lambda_{ij} F(c_{ij}) \geq \sum_{k \in K} \pi_k(\lambda) F(d^k)$.

Let us mention that this result is valid even when the values of the demands and the capacities are fractional.

Inequalities (23) are a generalization of the metric inequalities (22): if the function F is the identity function we get inequalities (22). Since F can be any superadditive function, there are more constraints to be satisfied by capacites and demands to allow feasibility of the single-path routing problem.

Let us consider, for example, a vector $\lambda = (\lambda_1, ..., \lambda_{|E|}) \in R_+^{|E|}$ induced by a cut $(B, X \setminus B)$ (*i.e.*, the set of edges having one end in B and another end in $X \setminus B$) where $B \subset X$. In other terms, $\lambda_{ij} = 1$ if $i \in B$ and $j \in X \setminus B$, and $\lambda_{ij} = 0$ otherwise. We already know that the inequality $\sum_{i \in B, j \in X \setminus B} c_{ij} \geq \sum_{o(k) \in B, s(k) \in X \setminus B} d^k$

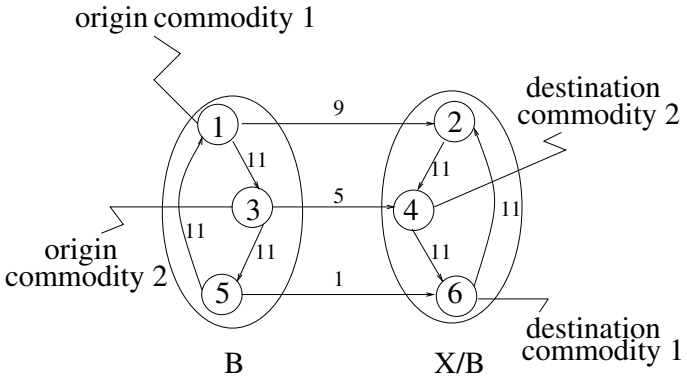

FIGURE 1. A violated superadditive semi-metric cut.

is a necessary condition for the multiple-path routing problem. According to Theorem 23, we should also have $\sum_{i\in B,j\in X\setminus B}(c_{ij})^2 \geq \sum_{o(k)\in B,s(k)\in X\setminus B}(d^k)^2$ and $\sum_{i\in B,j\in X\setminus B} c_{ij} \times \ln(c_{ij}) \geq \sum_{o(k)\in B,s(k)\in X\setminus B} d^k \times \ln(d^k)$.

EXAMPLE

The following example illustrates the necessary condition for single-path routing (Fig. 1). Let $R = (X, E, d)$ be a capacitated network, where $X = \{1, 2, 3, 4, 5, 6\}$ is the set of nodes. $E = \{(1, 2), (3, 4), (5, 6), (1, 3), (3, 5), (5, 1), (2, 4), (4, 6), (6, 2)\}$ is the set of arcs. The capacity values are $c_{12} = 9$, $c_{34} = 5$, $c_{56} = 1$. All other capacity values are fixed to 11. Commodity 1 sends 8 units from node 1 to node 6. Commodity 2 sends 7 units from node 3 to node 4.

There exist a nondecreasing and superadditive function F and a cut $(B, X \setminus B)$ such that $\sum_{i\in B,j\in X\setminus B} F(c_{ij}) < \sum_{o(k)\in B,s(k)\in X\setminus B} F(d^k)$. Indeed for $B = \{1, 3, 5\}$ and $F(x) = x^2$, we have $\sum_{i\in B,j\in X\setminus B} F(c_{ij}) = 9^2 + 5^2 + 1^2 = 107 < \sum_{o(k)\in B,s(k)\in X\setminus B} F(d^k) = 7^2 + 8^2 = 113$. In other terms, there is no way to route all demands using only one path.

Notice that this problem has a multiple-path routing solution. Commodity 1 sends 8 units through the path 1-2-4-6. Commodity 2 sends 5 units through the path 3-4, 1 unit through the path 3-5-6-2-4 and 1 unit through the path 3-1-2-4. Therefore the necessary condition for multiple-path routing $\sum_{i\in E,j\in X\setminus E} c_{ij} \geq \sum_{o(k)\in B,s(k)\in X\setminus B} d^k$ is satisfied for each cut $(B, X \setminus B)$.

Remarks

- As claimed by Theorem 4.1, superadditive semi-metric cuts are necessary to get an unsplittable feasible flow. However, they are generally not sufficient. Even if all the computational experiments of the next section will show that the gap between the linear relaxation and the integer problem

TABLE 1. Description of minimum cost single-path routing instances.

Instance	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Commodities	35	68	70	58	47	93	93	41	87	41	23	81	52	46
Nodes	14	24	29	18	19	27	23	28	24	19	14	26	29	20
Arcs	16	24	61	29	25	37	29	31	42	19	16	36	31	23

will be 0 after the addition of the superadditive cuts, this is due to the particular characteristics of the problem instances considered in this paper (taken from [7, 34]). The gap may be different from 0 for some other instances. Then, an interesting open question consists in determining some general classes of instances for which the superadditive semi-metric condition is sufficient to get a single-path routing.

- Theorem 4.1 can be strengthened in a trivial way using again duality. Let us consider r superadditive nondecreasing functions F_1,...,F_r. For each arc ij and each function F_u, consider a nonnegative variable λ_{ij}^u. Constraint 23 becomes

$$\sum_{u=1}^{r} \sum_{ij \in E} \lambda_{ij}^u F_u(c_{ij}) \geq \sum_{k \in K} \pi_k(\lambda)$$

while $\pi_k(\lambda)$ should satisfy

$$\pi_k(\lambda) \leq \sum_{e \in p} \sum_{u=1}^{r} \lambda_e^u F_u(d^k)$$

for each path $p \in P(k)$.

5. Preliminary computational experiments

We compare our cutting plane algorithms based on superadditive cuts (SCPA) and (QSCPA) to the algorithm proposed by Barnhart et al. [7] to solve the minimum cost single-path routing problem.

Barnhart *et al.* used an IBM RS6000/590 with CPLEX 3.0. They applied a branch-and-price-and-cut algorithm during 1 hour. In this paper, we use an Ultrasparc10 - 333 Mhz/128 Mo with CPLEX 7.1 [15].

The instances used in the experiments are described in Table 1. These instances were proposed by Parker and Ryan [34] and used by Barnhart *et al.* [7] to evaluate their algorithm. The number of commodities, the number of nodes and the number of arcs are reported for each instance.

Before presenting the results, let us give more details about our implementation.

Before applying either SCPA or QSCPA, some simple preprocessing is done. Indeed, if a demand value d^k is larger than the capacity value of a link c_e, then we know that commodity k can not use this link ($x_e^k = 0$).

As mentioned at the end of Section 2, instead of using the node-arc formulation, we will use a path formulation. This was also done in [7]. Columns are generated in a classical way by solving some shortest path problems (see, *e.g.*, [1, 7]).

Before starting the cutting plane algorithm, one path is initially associated with each commodity. Then, we generate columns to make the continuous relaxation of the problem feasible. When no more columns with negative reduced costs can be generated, we start our cutting plane algorithm. Given a current solution, we try to generate a superadditve cut for each knapsack constraint (each link). Then all the cuts are added to the linear relaxation which is solved again by column generation until no new columns can be generated. The same process is repeated until neither cuts nor paths can be generated. If the solution is not integer, a Branch&Bound should be used to find an integer solution. However, this will never be the case for all the instances of this paper: the solution was always integer at the end of the cutting plane algorithm.

First, let us focus on the results of SCPA and Barnhart *et al.* algorithm. The results are given in Table 2. For each instance, a first line gives the results of Barnhart *et al.* algorithm. Then follows a line that gives the results of SCPA (bold numbers). For each instance four values are provided. Column 2 gives the number of columns generated by each algorithm (including the number of paths that are initially included in the path-formulation). Column 3 reports the number of rows (inequalities) generated by each algorithm. The LP-IP gap is given in Column 4. Column 5 presents the running time.

The results can be divided into two categories. The first category includes instances 3, 6 and 9. The instances of this category are distinguished by a star. They are considered as being difficult by Barnhart *et al.* algorithm. Indeed the integer solution was not found in the allowed time (1 h). The second category contains all the other instances. They are considered as easy by Barnhart *et al.* algorithm. Indeed the integer solution was found in less than one hour.

Let us consider first the results for category 1. For instances 3, 6 and 9, SCPA generates a smaller number of columns and a smaller number of rows than Barnhart *et al.* algorithm.

We also notice that SCPA finds the optimal integer solution. The running time is quite small.

However, when comparing the branch-and-price-and-cut algorithm of [7] and SCPA, we should not focus on the computing time: the platforms used for computational experiments are different and the cplex version is more recent with SCPA.

Regarding the results for category 2, one can see again that SCPA generates a smaller number of columns and cuts than Barnhart *et al.* algorithm. Both SCPA and Barnhart's algorithm find the optimal integer solution in few seconds. Notice that the number of generated cuts is sometimes equal to 0. This means that the preprocessing combined with column generation are sufficient to find an optimal integer solution.

In conclusion, this first set of experiments shows the effectiveness of superadditive cuts to solve the single-path routing problem.

TABLE 2. Comparison between Barnhart's algorithm and SCPA (bolded numbers) on Parker and Ryan instances.

Instance	Columns	Rows	Gap (%)	CPU time (s)
1	180	7	0	0.5
1	**74**	**0**	**0**	**0.03**
2	295	3	0	0.88
2	**136**	**0**	**0**	**0.11**
3*	7378	835	3.63	3600
3*	**199**	**6**	**0**	**300**
4	418	88	0	7.41
4	**127**	**0**	**0**	**1.85**
5	232	9	0	0.63
5	**94**	**0**	**0**	**0.06**
6*	4345	728	2.16	3600
6*	**196**	**1**	**0**	**3.56**
7	493	154	0	23.11
7	**203**	**4**	**0**	**5.95**
8	206	8	0	0.84
8	**85**	**1**	**0**	**0.84**
9*	8186	751	1.16	3600
9*	**98**	**5**	**0**	**5.42**
10	190	23	0	1.11
10	**83**	**0**	**0**	**0.54**
11	106	7	0	0.31
11	**51**	**2**	**0**	**0.3**
12	430	54	0	4.25
12	**171**	**3**	**0**	**3.44**
13	253	7	0	0.74
13	**104**	**0**	**0**	**0.1**
14	248	37	0	2.36
14	**94**	**0**	**0**	**0.99**

The aim of the next experiments is to show that QSCPA can outperform SCPA. We focus again on Parker and Ryan instances. Results are given in Table 3. We give the results obtained when QSCPA is applied. For each instance, two values are provided: Column 2 gives the IP-LP gap and Column 3 presents the running time in seconds.

QSCPA was also able to find integer solutions for Parker and Ryan instances, including instances 3, 6 and 9. The running time is significantly improved for one of the most difficult instances. Indeed the running time on instance 3 decreases from 300 seconds with SCPA to 16 seconds with QSCPA. For other instances the running time is almost the same.

TABLE 3. Comparison between SCPA and QSCPA (bolded numbers) on Parker and Ryan instances.

Instance	GAP (%)	CPU time (s)
1	0	0.05
2	0	0.11
3*	0	16.31
4	0	1.76
5	0	0.04
6*	0	3.55
7	0	5.28
8	0	0.76
9*	0	4.83
10	0	0.51
11	0	0.29
12	0	3.4
13	0	0.09
14	0	0.8

6. Conclusions and further directions

Two superadditive cutting plane algorithms (SCPA) and (QSCPA) have been proposed in this paper to solve the minimum cost single-path routing problem. The first algorithm SCPA is based on superadditive cuts derived from Gomory theorem when applied to single knapsack constraints. Whereas the second algorithm QSCPA is based on superadditive cuts derived from the superadditive families proposed in the paper.

The running times and the quality of the solution obtained after the addition of these cuts are very promising.

The second algorithm QSCPA can generate superadditive cuts in a more efficient way without loosing too much in terms of constraints quality. The experimental results show that QSCPA can significantly outperform SCPA.

While both algorithms were used to solve only 14 problem instances, we think that the results obtained encourage the integration of these cuts into a branch-and-cut framework.

Notice that the superadditive step functions defined in this paper are valid for any knapsack problem. This clearly means that they can be tried to solve many other integer problems. We should also remember that the algorithm QSCPA is based on a very special kind of cuts. Many other superadditive functions can be considered and may lead to better results.

This paper presents also a simple necessary condition for the single-path routing problem. It is a generalization of the well known semi-metric condition of [23, 31]. Moreover, we think that these conditions may be sufficient for some classes of graphs and problem instances.

Finally, we may also try to generate more efficient constraints. Indeed, the cuts considered in this paper are sufficient to describe the convex hull of the integer solutions of a knapsack constraint and not the $0-1$ solutions. Said another way, to improve our algorithm we may generate some superadditive cuts integrating the fact that all variables must be in $[0,1]$. One way to do that is to use multiple-variable superadditive functions to define such cuts. Another possible direction is to still use single-variable functions but with combination of different constraints, in the spirit of Chvatal-Gomory rounding method.

Acknowledgements. The authors wish to thank the referees for a number of insightful comments that have led to a substantially improved paper.

References

[1] R.K. Ahuja, T.L. Magnanti and J.B. Orlin, *Network Flows.* Prentice-Hall (1993).
[2] F. Alvelos and J.M. Valério de Carvalho, Comparing Branch-and-price algorithms for the unsplittable multicommodity flow problem, in *Proceedings of the International Network Optimization Conference INOC, Evry-Paris, France* (2003) 7–12.
[3] C.A. Anderson, F. Fraughnaugh, M. Parker and J. Ryan, Path assignment for call routing: an application of Tabu search. *Ann. Oper. Res.* **41** (1993) 301–312.
[4] Y. Asano, *Experimental evaluation of approximation algorithms for the minimum cost multiple-source unsplittable flow problem.* ICALP Workshop (2000) 111–122.
[5] A. Atamtürk and D. Rajan, On splittable and unsplittable flow capacitated network-design arc-set polyhedra. *Math. Program.* **92** (2002) 315–333.
[6] G. Baier, E. Köhler and M. Skutella, The k-splittable flow problem. *Algorithmica* **42** (2005) 231–248.
[7] C. Barnhart, C.A. Hane and P.H. Vance, Using branch-and-price to solve origin-destination integer multicommodity flow problems. *Oper. Res.* **48** (2000) 318–326.
[8] C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsberg and P.H. Vance, Branch-and-price: column generation for solving huge integer programs. *Oper. Res.* **46** (1998) 316–329.
[9] W. Ben-Ameur, E. Gourdin, B. Liau and N. Michel, Routing strategies for IP networks. *In Telektronikk Magazine* **2/3** (2001) 145–158.
[10] W. Ben-Ameur, S. Besiktasliyan and B. Decocq, Optimal dimensioning of a ring. in *Proceeding of ITC17, Brazil* (2001).
[11] C.A. Burdet and E.L. Johnson, A subadditive approach to solve linear integer programs. *Ann. Discrete Math.* **1** (1977) 117–144.
[12] F. Chauvet, P. Chrétienne, P. Mahey and B. Vatinlen, Minimisation du nombre de chemins décomposant un flot, *in Proceeding of Algotel* (2004).
[13] D. Coudert and H. Rivano, Lightpath assignment for multifibers WDM networks with wavelength translators. *Proceedings of the Global Telecommunications Conference* (2002) 2686–2690.
[14] T.G. Crainic, A. Frangioni and B. Gendron, Bundle-based relaxation methods for multicommodity capacitated fiwed charge network design. *Discrete Appl. Math.* **112** (2001) 73–99.
[15] CPLEX Optimization, Inc. Using the CPLEX Callabe Library and CPLEX Mixed Integer Library, version 7.1. (2001).
[16] G. Dahl, A. Martin and M. Stoer, Routing through virtual paths in layered telecommunication networks. *Oper. Res.* **47** (1999) 693–702.
[17] Y. Dinitz, N. Garg and M.X. Goemans, On the single-source unsplittable flow problem. *Combinatorica* **19** (1999) 1–25.

[18] J. Geffard, A 0-1 model for singly routed traffic in telecommunications. *Ann. Telecom.* **56** (2001) 140–149.
[19] R.E. Gomory, E.L. Johnson and L. Evans, Corner polyhedra and their connection with cutting planes. *Math. Program.* **96** (2003) 321–339.
[20] Z. Gu, G.L. Nemhauser and M.W.P. Savelsbergh, Cover inequalities for 0-1 linear programs: computation. *INFORMS J. Comput.* **10** (1998) 427–437.
[21] Z. Gu, G.L. Nemhauser and M.W.P. Savelsbergh, Cover inequalities for 0-1 linear programs: complexity. *INFORMS J. Comput.* **11** (1999) 117–123.
[22] K. Holmberg and D. Yuan, A Lagrangian heuristic based branch-and-bound approach for the capacitated network design problem. *Oper. Res.* **48** (2000) 461–481.
[23] M. Iri, On an extension of the maximum-flow minimum-cut theorem to multicommdity flows. *J. Oper. Res. Soc. Japan* **13** (1971) 129–135.
[24] J.M. Kleinberg, *Approximation algorithms for disjoint path problems*. Ph.D. dissertation, M.I.T. (1996).
[25] S.G. Kolliopoulos and C. Stein, Approximation algorithms for single-source unsplittable flow. *SIAM J. Comp.* **31** (2002) 919–946.
[26] P. Kolman, A note on the greedy algorithm for the unsplittable flow problem. *Information Processing Lett.* **88** (2003) 101–105.
[27] M. Laguna and F. Glover, Bandwidth packing: a tabu search approach. *Manag. Sci.* **39** (1993) 492–500.
[28] D. Lorenz, A. Orda, D. Raz and Y. Shavitt, *How good can IP routing be?* DIMACS Technical Report 2001-17, May 2001.
[29] M.E. Lübbecke and J. Desrosiers, Selected Topics in Column Generation. *Oper. Res.* **53** (2005) 1007–1023.
[30] G.L. Nemhauser and L.A. Wolsey, *Integer and combinatorial optimization*. Wiley & Sons (1988).
[31] K. Onaga and O. Kakusho, On feasibility conditions of multicommodity flows in networks. *IEEE Tran. Circuit Theory* **4** (1971) 425–429.
[32] K. Park, S. Kang and S. Park, An integer programming approach to the bandwidth packing problem. *Manage. Sci.* **42** (1996) 1277–1291.
[33] S. Park, D. Kim and K. Lee, An Integer Programming Approach to the Path Selection Problems. *Proceedings of the International Network Optimization Conference INOC, Evry-Paris, France* (2003) 448–453.
[34] M. Parker and J.M. Ryan, A column generation algorithm for bandwidth packing. *Telecom. Syst.* **2** (1993) 185–195.
[35] A. Schriver, P. Seymour and P. Winkler, The ring loading problem. *SIAM J. Discrete Math.* **11** (1998) 1–14.
[36] M. Skutella, Approximating the single-source unsplittable min-cost flow problem. *Math. Program. Ser. B* **91** (2002) 493–514.
[37] Y. Wang and Z. Wang, Explicit Routing Algorithms for Internet Traffic Engineering, in *Proceedings of the International Conference on Computer Communication Networks, Boston, USA* (1999).
[38] W.E. Wilhelm, A technical review of column generation in integer programming. *Optim. Eng.* **2** (2001) 159–200.
[39] L.A. Wolsey, Valid inequalities and superadditivity for 0-1 integer programs. *Math. Oper. Res.* **2** (1977) 66–77.